# High-Order Convergence of Spectral Deferred Correction Methods on General Quadrature Nodes

Tao Tang*, Hehu Xie †, and Xiaobo Yin‡

**Abstract**

It has been demonstrated that spectral deferred correction (SDC) methods can achieve arbitrary high order accuracy and possess good stability properties. There have been some recent interests in using high-order Runge-Kutta methods in the prediction and correction steps in the SDC methods, and higher order rate of convergence is obtained provided that the quadrature nodes are *uniform*. The assumption of the use of uniform mesh has a serious practical drawback as the well-known Runge phenomenon may prevent the use of reasonably large number of quadrature nodes. In this work, we propose a modified SDC methods with high-order integrators which can yield higher convergence rates on *both* uniform and non-uniform quadrature nodes. The expected high-order of accuracy is theoretically verified and numerically demonstrated.

## 1 Introduction

In [4], a spectral deferred correction (SDC) method was proposed by coupling the Gaussian quadrature with Picard iterations, which was shown to possess very high-order of accuracy and larger stability region. The success of the SDC is due to several issues. Firstly, the difficulty of equidistant polynomial interpolation is eliminated by using the Gaussian quadrature on each subinterval. Secondly, the instability of numerical differentiation is eliminated by solving the equivalent Picard integral equations rather than directly solving the ODEs. Furthermore, the Picard process relaxes the restriction of the uniform distribution of the quadrature nodes required by the classical deferred correction methods.

*Department of Mathematics, Hong Kong Baptist University, Hong Kong, China(ttang@hkbu.edu.hk)

†LSEC, NCMIS , Institute of Computational Mathematics, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China(hhxie@lsec.cc.ac.cn)

‡School of Mathematics and Statistics, Cental China Normal University, Wuhan, 430079, China(yinxb@mail.ccnu.edu.cn)

The low-order methods such as the forward Euler or backward Euler methods are employed [4] during the correction loops, while high-order integrators were investigated by Christlieb et al. [2] where an $r^{th}$-order Runge-Kutta (RK) integrator is used to solve the error equation in the correction loop. With the same number of function evaluations, it was shown that the use of the higher-order integrators can have one magnitude of accuracy improvement. However, the order increasement in the correction loop does not hold for non-uniform grids, e.g., the Gaussian quadrature nodes are excluded. The aim of this paper is to have a better understanding of the convergence-rate issues on *non-uniform quadrature nodes*, in particular we wish to consider the case of using SDC methods with high-order integrators during correction loops. More precisely, we consider how to recover high-order rate of the convergence for higher-order integrators with non-uniform quadrature nodes. To this end, we will establish a recursive relation for error functions and then give the convergence analysis. Then we make a simple modification in the correction step to improve the regularity of the error functions; this will allow us to extend high order convergence behaviors to the non-uniform quadrature nodes.

The rest of the paper is arranged as follows. In Section 2, we review the SDC methods and discuss their error analysis. We will also present a modified SDC strategy to extend high order convergence behaviors to the non-uniform quadrature nodes. In Section 3, numerical experiments will be reported to support the theoretical predictions. The superior stability properties of the modified SDC methods will be demonstrated in Section 4. The final section gives some concluding remarks.

## 2 The SDC method and its convergence

For simplicity, we consider an IVP consisting of a scalar ODE and initial condition

$$\left\{ \begin{array}{rcl} y'(t) & = & f(t, y(t)), \qquad t \in (a, b], \\ y(a) & = & y_a. \end{array} \right. \tag{2.1}$$

We assume the function $f : [a, b] \times \mathbb{R} \mapsto \mathbb{R}$ satisfies the following Lipschitz continuous condition

$$|f(\bullet, y_1) - f(\bullet, y_2)| \leq L|y_1 - y_2|, \tag{2.2}$$

with the Lipschitz constant $L$. To implement the SDC methods, we discretize the time domain $[a, b]$ into intervals to get a mesh

$$\mathcal{T}_h := \big\{t_n : \ a = t_0 < t_1 < \cdots < t_N = b\big\}.$$

We set $I_n := (t_n, t_{n+1}]$, $\bar{I}_n := [t_n, t_{n+1}]$, $h_n := t_{n+1} - t_n$, with $n = 0, 1, \cdots, N - 1$. The quantity

$$h := \max\{h_n : \ 0 \leq n \leq N - 1\}$$

will be called the size of the mesh $\mathcal{T}_h$. Let $X_h$ be the grid points which is given by

$$X_h := \Big\{ t_{n,i} := t_n + c_i h_n : 0 \le c_1 < \cdots < c_m \le 1 \Big\}, \tag{2.3}$$

where $\{c_i\}$ is a prescribed set of collocation parameters; for a given mesh $\mathcal{T}_h$ the collocation parameters completely determine $X_h$. It is noted that the classical deferred correction methods are based on the equispaced nodes in each interval $I_n$, i.e., $c_i = i/m$.

In this section, we will concentrate on the subinterval $I_n$, so in most cases the superscript $n$ will be dropped (e.g., $X_h$ defined by (2.3) also depends on $n$). We introduce the standard Lagrange interpolation operator $L_m$ defined on the interval $I_n$:

$$L_m(\overrightarrow{\varphi}) \;=\; \sum_{i=1}^{m} \varphi_i l_{n,i}(t) \tag{2.4}$$

for vectors $\overrightarrow{\varphi} = [\varphi_1, \cdots, \varphi_m]^T \in \mathbb{R}^m$, where the function $l_{n,i}(t)$ denotes the corresponding Lagrange basis functions associated with the collocation points $\{t_{n,i}\}_{i=1}^m$.

## 2.1 The SDC methods

This subsection reviews the SDC methods proposed in [4] for (2.1). The SDC method is based on the following equivalent integral equation associated with (2.1)

$$y(t) = y_a + \int_a^t f(s, y(s))ds, \quad t \in (a, b]. \tag{2.5}$$

In the collocation method, the solution of (2.5) is approximated by an element $u_h$ of the piecewise polynomial space

$$S_{m-1}^{(-1)}(\mathcal{T}_h) := \Big\{ v \in C^{-1}([a,b]) : \; v|_{\bar{I}_n} \in \mathcal{P}_{m-1} \; (0 \le n \le N-1) \Big\}, \tag{2.6}$$

where $\mathcal{P}_{m-1}$ denotes the space of all (real) polynomials of degree not exceeding $m-1$. The collocation solution $u_h \in S_{m-1}^{(-1)}(\mathcal{T}_h)$ for (2.5) is defined by the collocation equation

$$u_h(t) = y_a + \int_a^t f(s, u_h(s))ds, \quad t \in X_h. \tag{2.7}$$

Let

$$\overrightarrow{Y}_m = [u_h(t_{n,1}), \cdots, u_h(t_{n,m})]^T, \quad \overrightarrow{y}_m = [y(t_{n,1}), \cdots, y(t_{n,m})]^T, \tag{2.8}$$

where $c_j$ is defined by (2.3) and $y$ is the exact solution of (2.1). It is known that the following error estimate for the collocation approximation $Y_m$ holds (c.f. [1, Theorem 2.2.3])

$$\|\overrightarrow{y}_m - \overrightarrow{Y}_m\| \leq Ch^m \|y\|_{m+1}, \tag{2.9}$$

where $\| \bullet \|_{m+1}$ denotes the maximum norm of $y^{(m+1)}$.

Denote the operator $K$ by the following functional

$$Kf(y) = \int_{t_n}^{t} f(s, y(s))ds, \quad \text{for } t \in (t_n, t_{n+1}]. \tag{2.10}$$

It follows from (2.1) that in $I_n$

$$y(t) = y(t_n) + Kf(y), \qquad t \in [t_n, t_{n+1}]. \tag{2.11}$$

Define a matrix operator $K_m$ corresponding to the collocation parameters $\{c_i\}$ by

$$K_m f(\overrightarrow{\eta}) \;\; = \;\; \left( \sum_{j=1}^{m} f(t_{n,j}, \eta_j) \int_{t_n}^{t_{n,i}} l_{n,j}(s)ds \right)_{1\leq i\leq m} \tag{2.12}$$

for the vector $\overrightarrow{\eta}$ and $K_m \overrightarrow{\eta} \in \mathbb{R}^m$. It follows from the Lipschitz continuous condition (2.2) that

$$\|K_m f(\overrightarrow{\eta}_1) - K_m f(\overrightarrow{\eta}_2)\| \leq Ch \|\overrightarrow{\eta}_1 - \overrightarrow{\eta}_2\|, \tag{2.13}$$

where (and in the rest of the paper) $\|\bullet\|$ denotes the standard $l^\infty$-vertor norm. Using the definition (2.12), the collocation approximation $Y_m$ for (2.1) can be written as

$$\overrightarrow{Y}_m = \overrightarrow{u}_n + K_m f(\overrightarrow{Y}_m) \tag{2.14}$$

with $\overrightarrow{u}_n = [u_n, \cdots, u_n] \in \mathbb{R}^m$.

The algorithm of the SDC method in the interval $[t_n, t_{n+1}]$ is given below.

**Algorithm 2.1.** *(SDC method):*

*1 (Prediction).*

*Use a $k_0$-th order numerical method to compute an initial approximation over the grid nodes (2.3), $\overrightarrow{\eta}^{[0]} = \left[\eta_1^{[0]}, \cdots \eta_m^{[0]}\right]^T$ which is a $k_0$-th order approximation to $\overrightarrow{y}_m$ defined by (2.8).*

*2 (Correction).*

*For $j = 1, \cdots, J$, do*

- *2(a). Compute the interpolating polynomial $L_m\left(\overrightarrow{\eta}^{[j-1]}\right)$.*

- *2(b). Compute the residual for* $\overrightarrow{\eta}^{[j-1]}$
$$\overrightarrow{\varepsilon}^{[j-1]} = \overrightarrow{u}_n + K_m f(\overrightarrow{\eta}^{[j-1]}) - \overrightarrow{\eta}^{[j-1]}. \tag{2.15}$$
*Define* $\varepsilon^{[j-1]} = L_m(\overrightarrow{\varepsilon}^{[j-1]})$ *and define the error function for* $\overrightarrow{\eta}^{[j-1]}$
$$\tilde{e}^{[j-1]}(t) = y_m(t) - L_m\big(\overrightarrow{\eta}^{[j-1]}\big). \tag{2.16}$$

- *2(c). Form the error equation*
$$\tilde{e}^{[j-1]}(t) = K_m f\big((\tilde{e}^{[j-1]}) + L_m(\overrightarrow{\eta}^{[j-1]})\big) - K_m f(\overrightarrow{\eta}^{[j-1]}) + \varepsilon^{[j-1]}(t), \tag{2.17}$$
*for* $t \in (t_n, t_{n+1}]$, *with* $\tilde{e}^{[j-1]}(t_n) = 0$.

- *2(d). Use an k-th order method to compute an approximate solution* $\delta_i^{[j-1]} \approx \tilde{e}^{[j-1]}(t_{n,i})$ *to the error equation at the grid points* $t_{n,i}$ *on* $[t_n, t_{n+1}]$.

- *2(e). Define a new approximate solution* $\overrightarrow{\eta}^{[j]} = \overrightarrow{\eta}^{[j-1]} + \overrightarrow{\delta}^{[j-1]}$.

There are various implementation details for the SDC methods, see, e.g., discussions on the distribution of grid points (2.3) in [7], the choice of numerical integrations, or different numerical methods in the prediction and correction steps in [6, 8].

## 2.2 Convergence analysis

We first illustrate how the convergence order is improved at each correction step. Then adding the error estimates for the prediction step, the error estimate for the SDC methods will be obtained. Note the convergence result in this subsection is not new. For example, the convergence for the SDC method on uniform quadrature nodes with high-order RK correction steps was established in [2] using a *smoothness of rescaled error vector* approach. However, the new framework given in this subsection will allow us to further improve the SDC algorithm as to be demonstrated in the following subsection.

To implement SDC methods, a lower order numerical method such as forward Euler, backward Euler, and second-order RK method is applied to obtain approximations of the error equations in the correction steps. Similar to the definition of integral operator $K_m$ to (2.14), we also define the lower-order discrete integral operator $\widetilde{K}_m$ corresponding to the numerical methods such as forward Euler, backward Euler, and second-order RK methods. It follows from the Lipschitz condition (2.2) that

$$\|\widetilde{K}_m f(\overrightarrow{\eta}_1) - \widetilde{K}_m f(\overrightarrow{\eta}_2)\| \leq Ch\|\overrightarrow{\eta}_1 - \overrightarrow{\eta}_2\|, \tag{2.18}$$

where $C$ is a constant dependent on $L$ but independent of the time step size $h$.

**Theorem 2.1.** *Assume the solution $y(t)$ of (2.1) is $(m+1)$-times continuously differentiable. Let $\overrightarrow{\eta}^{[J]}$ be computed in the Correction step of Algorithm 2.1. If the stepsize $h$ is sufficiently small, then the following error estimate holds*

$$\|\overrightarrow{y}_m - \overrightarrow{\eta}^{[J]}\| \le Ch^{k_0+J}\|y\|_{k_0+1} + Ch^m\|y\|_{m+1}, \tag{2.19}$$

*where $\overrightarrow{y}_m$ is defined by (2.8), the constant $C$ is independent of $h$, $k_0$ is the convergence order for numerical schemes used in the prediction step, and $J$ is the number of Correction cycles in Algorithm 2.1.*

*Proof.* It follows from Algorithm 2.1 that

$$\begin{aligned} \overrightarrow{\delta}^{[j-1]} &= \widetilde{K}_m f\big(\overrightarrow{\delta}^{[j-1]} + \overrightarrow{\eta}^{[j-1]}\big) - \widetilde{K}_m f(\overrightarrow{\eta}^{[j-1]}) + \overrightarrow{\varepsilon}^{[j-1]} \\ &= \widetilde{K}_m f(\overrightarrow{\eta}^{[j]}) - \widetilde{K}_m f(\overrightarrow{\eta}^{[j-1]}) + \overrightarrow{u}_n + K_m f(\overrightarrow{\eta}^{[j-1]}) - \overrightarrow{\eta}^{[j-1]} \\ &= \widetilde{K}_m f(\overrightarrow{\eta}^{[j]}) - \widetilde{K}_m f(\overrightarrow{\eta}^{[j-1]}) + \overrightarrow{Y}_m - K_m f(\overrightarrow{Y}_m) \\ &\quad + K_m f(\overrightarrow{\eta}^{[j-1]}) - \overrightarrow{\eta}^{[j-1]}, \end{aligned} \tag{2.20}$$

where in the last step we have used (2.14). Using the above result gives

$$\begin{aligned} \overrightarrow{\eta}^{[j]} &= \overrightarrow{\eta}^{[j-1]} + \overrightarrow{\delta}^{[j-1]} \\ &= \widetilde{K}_m f(\overrightarrow{\eta}^{[j]}) - \widetilde{K}_m f(\overrightarrow{\eta}^{[j-1]}) + \overrightarrow{Y}_m - K_m f(\overrightarrow{Y}_m) \\ &\quad + K_m f(\overrightarrow{\eta}^{[j-1]}). \end{aligned} \tag{2.21}$$

Consequently, we have

$$\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]} = \widetilde{K}_m f(\overrightarrow{\eta}^{[j-1]}) - \widetilde{K}_m f(\overrightarrow{\eta}^{[j]}) + K_m f(\overrightarrow{Y}_m) - K_m f(\overrightarrow{\eta}^{[j-1]}). \tag{2.22}$$

This, together with (2.13) and (2.18), yield

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| \le Ch\Big(\|\overrightarrow{\eta}^{[j-1]} - \overrightarrow{\eta}^{[j]}\| + \|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j-1]}\|\Big), \tag{2.23}$$

which gives

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| \le Ch\Big(\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| + \|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j-1]}\|\Big). \tag{2.24}$$

If $h$ is sufficiently small, the above result gives the following recurence formula

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| \le Ch\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j-1]}\|, \tag{2.25}$$

which yields

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[J]}\| \le Ch^J\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[0]}\|. \tag{2.26}$$

Note that

$$\begin{aligned} \|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[0]}\| &\le \|\overrightarrow{Y}_m - \overrightarrow{y}_m\| + \|\overrightarrow{\eta}^{[0]} - \overrightarrow{y}_m\|_{0,\infty} \\ &\le Ch^m\|y\|_{m+1} + Ch^{k_0}\|y\|_{k_0+1}, \end{aligned} \tag{2.27}$$

where in the last step we have used (2.9). The desired estimate (2.19) follows from (2.9), (2.26) and (2.27). □

## 2.3 Convergence enhancement on non-uniform nodes

Since the correction step in Algorithm 2.1 includes only one Picard integration, the accuracy can only be improved by one order after each correction step. It is also interresting to observe in [6, 7, 8] that on non-uniform quadrature nodes even high order numerical methods are applied in Step 2(e) for solving the error equation only one order accuracy improvement can be obtained with each correction step. On the other hand, the classical deferred correction methods are based on the uniform distribution of grid points (see, e.g., [9, 10]), which is also a necessary condition for accuracy enhancement at the correction steps in the SDC methods (see, e.g., [3]).

In this subsection, we will modify the correction steps so that higher order enhancement can be obtained on non-uniform nodes. The key idea is to improve the regularity of the error function so that higher order numerical methods used in Step 2(e) can yield high accuracy. The key point is that the regularity of the error function can be improved by one order after each Picard integration even on the non-uniform quadrature nodes (see [11]).

**Algorithm 2.2.** *(Modified SDC method):*
*Same as Algorithm 2.1, except that before doing the step 2(a) compute an improved* $\overrightarrow{\eta}^{[j-1]}$*:*

$$\overrightarrow{\xi}^{[\mu]} = \overrightarrow{u}_n + K_m f(\overrightarrow{\xi}^{[\mu-1]}), \quad \mu = 1, \cdots, k-1, \tag{2.28}$$

$$\overrightarrow{\xi}^{[0]} = \overrightarrow{\eta}^{[j-1]}. \tag{2.29}$$

*After the above* $k-1$ *iterations, let* $\overrightarrow{\eta}^{[j-1]} = \overrightarrow{\xi}^{[k-1]}$.

Note in the above modified algorithm, the number of iteration cycles in (2.28) is $k-1$, while the order of numerical methods used to approximate the error equations in Step 2(e) is $k$, see Algorithm 2.1. The reason for doing this will be seen in the proof of the following theorem.

**Theorem 2.2.** *Assume the solution* $y(t)$ *of (2.1) is* $(m+1)$*-times continuously differentiable. Let* $\overrightarrow{\eta}^{[J]}$ *be computed in the Correction step of the SDC Algorithm 2.2. If the stepsize* $h$ *is sufficiently small, then the following error estimate holds*

$$\|\overrightarrow{y}_m - \eta^{[J]}\| \leq Ch^{Jk+k_0}\|y\|_{k_0+1,\infty} + Ch^m\|y\|_{m+1,\infty}, \tag{2.30}$$

*where the constant* $C$ *is independent of* $h$*,* $k_0$ *is the order of convergence for numerical schemes used in the Prediction step,* $J$ *is the number of Correction cycles in Algorithm 2.1, and* $k$ *is the number occur in (2.28) and in Step 2(d) of Algorithm 2.1. Furthermore, if the* $m$*-quadrature points in (2.3) are chosen to be Legendre-Gauss points, then we have*

$$\|\overrightarrow{y}_m - \eta^{[J]}\| \leq Ch^{Jk+k_0}\|y\|_{k_0+1,\infty} + Ch^{2m}\|y\|_{2m+1,\infty}. \tag{2.31}$$

*Proof.* Without lose of generality, we assume $k \geq 2$. It follows from (2.28) and (2.14) that

$$\overrightarrow{\xi}^{[\mu]} = \overrightarrow{Y}_m - K_m f(\overrightarrow{Y}_m) + K_m f(\overrightarrow{\xi}^{[\mu-1]}), \tag{2.32}$$

which leads to

$$\|\overrightarrow{\xi}^{[\mu]} - \overrightarrow{Y}_m\| \leq Ch\|\overrightarrow{\xi}^{[\mu-1]} - \overrightarrow{Y}_m\|, \tag{2.33}$$

and consequently

$$\|\overrightarrow{\xi}^{[k-1]} - \overrightarrow{Y}_m\| \leq Ch^{k-1}\|\overrightarrow{\xi}^{[0]} - \overrightarrow{Y}_m\| = Ch^{k-1}\|\overrightarrow{\eta}^{[j-1]} - \overrightarrow{Y}_m\|. \tag{2.34}$$

Note this $\overrightarrow{\xi}^{[k-1]}$ is a modification of the previous $\overrightarrow{\eta}^{[j-1]}$; this new $\overrightarrow{\xi}^{[k-1]}$ will be used in Steps 2(a)-2(e). Hence combining (2.34) and (2.25) gives

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| \leq Ch^k\|\overrightarrow{\xi}^{[0]} - \overrightarrow{Y}_m\|,$$

which together with (2.29) yields

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[j]}\| \leq Ch^k\|\overrightarrow{\eta}^{[j-1]} - \overrightarrow{Y}_m\|. \tag{2.35}$$

Consequently, we have

$$\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[J]}\| \leq Ch^{Jk}\|\overrightarrow{Y}_m - \overrightarrow{\eta}^{[0]}\|. \tag{2.36}$$

The remaining proof for (2.30) is the same as that in Theorem 2.1. The result of (2.31) follows directly from (2.27) and the property of the Legendre-Gauss points. □

# 3 Numerical experiments

In this section, we use some numerical examples to illustrate the order of accuracy for SDC methods with different distribution of quadrature nodes. Consider the initial value problem:

$$y' = y + \cos(t+1)e^{t+1}, \quad t \in (-1, 1], \tag{3.1}$$

$$y(-1) = 1. \tag{3.2}$$

The exact solution is $y(t) = (1 + \sin(t+1))e^{t+1}$. In this section, the "steps" in each result table means that the time step size is $\frac{1}{\text{steps}}$ (c.f. [2]).

Table 1: Errors and convergence rates for the SDC method on 7 *uniform* quadrature nodes. RK2 is used for prediction and correction ($k_0 = k = 2$).

| steps | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| 1 RK2 ($J = 0$) | 1.64E-02 | 4.17E-03 | 1.87E-03 | 1.05E-03 |
| order | – | 1.97 | 1.98 | 1.99 |
| 2 RK2 ($J = 1$) | 1.39E-05 | 8.23E-07 | 1.60E-07 | 5.00E-08 |
| order | – | 4.08 | 4.05 | 4.03 |
| 3 RK2 ($J = 2$) | 1.33E-08 | 1.87E-10 | 1.58E-11 | 2.74E-12 |
| order | – | 6.15 | 6.10 | 6.07 |

## 3.1 Accuracy vs nodes distributions

We first check the accuracy improvement for the SDC methods with high-order correction steps. Here, we solve (3.1)-(3.2) with a RK2 prediction and correction steps on seven uniformly distributed quadrature nodes. The corresponding results are shown in Table 1. As in [2] and [5], each RK2 correction step can increase two orders of accuracy. This is slightly better than the theoretical results in Theorem 2.1, i.e., one extra order is gained in each correction step. This 'superconvergence' behavior is related with the use of the uniform nodes, which is quite typical for the prediction-correction type methods.

Table 2: Errors and convergence orders for the SDC method on 9 *non-uniform* quadrature nodes of the form (3.3). RK2 is used for prediction and correction ($k_0 = k = 2$).

| steps | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| 1 RK2 ($J = 0$) | 1.52E-02 | 4.02E-03 | 1.82E-03 | 1.03E-03 |
| order | – | 1.92 | 1.95 | 1.97 |
| 2 RK2 ($J = 1$) | 2.76E-05 | 2.73E-06 | 7.36E-07 | 2.95E-07 |
| order | – | 3.33 | 3.23 | 3.18 |
| 3 RK2 ($J = 2$) | 6.35E-08 | 2.30E-09 | 3.56E-10 | 9.80E-11 |
| order | – | 4.79 | 4.60 | 4.49 |

To show the result of Theorem 2.1 is sharp, we apply the SDC methods on *non-uniform* quadrature nodes. We first consider quadrature nodes with linearly increasing spacing. More precisely, 9 points (i.e., $m = 9$) in each interval satisfying

$$t_{n,i} - t_{n,i-1} = i\bar{h}, \quad \bar{h} = \frac{2h}{m(m+1)}, \quad i = 1, \cdots, m \tag{3.3}$$

are used, where $h = t_{n+1} - t_n$ denotes the interval size, $t_{n,0} = t_n$ and $t_{n,m} = t_{n+1}$. RK2 is then applied in the prediction and correction steps. The results are shown in Table 2, which indicates that the convergence order can be improved by only one after each correction step. In other words, convergence order improvement is decreased due to the use of non-uniform quadrature nodes.

To see the difference of the modified SDC method, Algorithm 2.2 is employed to solve (3.1)-(3.2) using the non-uniform nodes with RK2 in the prediction and correction steps. Tables 3 and 4 give the numerical results obtained by using the non-uniform grid (3.3) and 9 Chebyshev-Lobatto quadrature nodes, respectively, where the rate of convergence predicted by Theorem 2.2, i.e., $k_0 + Jk$ is confirmed.

Finally, to verify (2.31), we use 4 Legendre-Gauss quadrature nodes with RK2 in the prediction and correction steps. Table 5 gives the numerical results, which confirm that convergence order of 8 can be arrived after 3 correction steps, and $m = 4$ is sufficient to balance $k_0 + Jk = 8$ and $2m = 8$ in (2.31).

## 3.2 Accuracy vs integrators

We consider the IVP

$$y' = -2\pi \sin(2\pi t) - 2(y - \cos(2\pi t)), \quad t \in (0, 20], \tag{3.4}$$

$$y(0) = 1, \tag{3.5}$$

which has the exact solution $y = \cos(2\pi t)$. This test example was used in [2].

We solve (3.4)-(3.5) using Algorithm 2.2 with 8 Chebyshev-Lobatto nodes and various integrators. Specifically, given the number of intervals $N$, the SDC method is carried out in each interval $[t_{i-1}, t_i]$ (with $t_i - t_{i-1} = 20/N$, $i = 1, 2, \cdots, N$) with 5 Legendre-Gauss nodes. Let SDC-FE denote the SDC method with the forward Euler integrator as the predictor and corrector. SDC-RK2 and SDC-RK4 are defined similarly.

It is observed from Table 6 that the numerical errors obtained by using only 5 Legendre-Gauss quadrature nodes in Algorithm 2.2 are consistently one order of magnitude smaller than those obtained by using 8 uniform quadrature nodes in the SDC method in [2]. Moreover, after comparing the results with [2], we also find that the errors associated with the Legendre-Gauss quadrature nodes are smaller than those obtained by using the uniform quadrature nodes. The corresponding efficiency comparison is demonstrated in Figure 1, which shows that the modified SDC method on the Legendre-Gauss nodes has better efficiency and the use of the specific non-uniform meshes can enhance accuracy.

A final efficiency comparison will be made by considering error against the numbers of function evaluations. Here function evaluation means the computation of function $f$ in (2.1). The performance of the uniformly distributed nodes and the Legendre-Gauss quadrature nodes is compared in Figure 3.2. It is observed from

Table 3: Same as Table 2, except using Algorithm 2.2 ($k_0 = k = 2$).

| steps | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| 1 RK2 ($J = 0$) | 1.52E-02 | 4.02E-03 | 1.82E-03 | 1.03E-03 |
| order | – | 1.92 | 1.96 | 1.97 |
| 2 RK2($J = 1$) | 5.42E-06 | 3.02E-07 | 5.70E-08 | 1.76E-08 |
| order | – | 4.17 | 4.11 | 4.08 |
| 3 RK2 ($J = 2$) | 1.90E-09 | 2.37E-11 | 1.99E-12 | 2.17E-13 |
| order | – | 6.33 | 6.11 | 7.70 |

Table 4: Same as Table 2, except using Algorithm 2.2 on 9 Chebyshev-Lobatto nodes ($k_0 = k = 2$).

| steps | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| 1 RK2 ($J = 0$) | 1.48E-02 | 3.79E-03 | 1.69E-03 | 9.56E-04 |
| order | – | 1.97 | 1.98 | 1.99 |
| 2 RK2 ($J = 1$) | 4.73E-06 | 2.47E-07 | 4.56E-08 | 1.39E-08 |
| order | – | 4.26 | 4.17 | 4.13 |
| 3 RK2 ($J = 2$) | 1.44E-09 | 1.64E-11 | 1.27E-12 | 2.11E-13 |
| order | – | 6.46 | 6.31 | 6.24 |

Table 5: Same as Table 3, except on 4 Legendre-Gauss quadrature nodes ($k_0 = k = 2$).

| steps | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| 1 RK2 ($J = 0$) | 4.30E-02 | 1.11E-02 | 5.01E-03 | 2.84E-03 |
| order | – | 1.95 | 1.97 | 1.98 |
| 2 RK2 ($J = 1$) | 3.69E-05 | 2.93E-06 | 6.23E-07 | 2.04E-07 |
| order | – | 3.65 | 3.82 | 3.88 |
| 3 RK2 ($J = 2$) | 3.34E-08 | 8.41E-10 | 8.43E-11 | 1.60E-11 |
| order | – | 5.31 | 5.67 | 5.78 |
| 4 RK2 ($J = 3$) | 1.25E-09 | 4.95E-12 | 1.87E-13 | 1.95E-14 |
| order | – | 7.98 | 8.09 | 7.84 |

Figure 3.2 that the modified SDC method with special non-uniform nodes can gain better efficiency than the original SDC method given in [2, 4].

# 4 Concluding remarks

To conclude this paper, we first comment the stability impact of Algorithms 2.1 and 2.2. Consider

$$\left\{ \begin{array}{rcl} y'(t) & = & \lambda y(t), \\ y(0) & = & 1, \end{array} \right. \tag{4.1}$$

Table 6: Numerical accuracy for (3.4)-(3.5) at $T = 20$ obtained by using Algorithm 2.2 and 5 Legendre-Gauss quadrature nodes; the corresponding results in brackets are obtained in [2] using 8 uniformly distributed quadrature nodes. Here 'nfv' stands for number of function evaluations of Algorithm 2.2.

| – | SDC-FE<br>($k_0 = k = 1, J = 7$) | | SDC-RK2<br>($k_0 = k = 2, J = 3$) | | SDC-RK4<br>($k_0 = k = 4, J = 1$) | |
|---|---|---|---|---|---|---|
| steps(N) | error | order<br>nfv | error | order<br>nfv | error | order<br>nfv |
| 40 | 6.38E-08<br>(5.47E-06) | –<br>3040 | 9.64E-08<br>(5.48E-06) | –<br>3480 | 7.31E-08<br>(4.49E-07) | –<br>3160 |
| 80 | 4.36E-11<br>(1.49E-08) | 10.51<br>6080 | 8.43E-11<br>(1.49E-08) | 10.16<br>6960 | 3.31E-11<br>(1.17E-09) | 11.11<br>6320 |
| 120 | 2.32E-12<br>(5.42E-10) | 7.23<br>9120 | 1.68E-12<br>(5.43E-10) | 9.65<br>10440 | 1.55E-12<br>(4.27E-11) | 7.56<br>9480 |
| 160 | 3.09E-13<br>(5.30E-11) | 7.04<br>12160 | 1.19E-13<br>(5.31E-11) | 9.20<br>13920 | 2.74E-13<br>(4.16E-12) | 6.01<br>12640 |
| 200 | 6.47E-14<br>(8.79E-12) | 7.00<br>15200 | 1.94E-14<br>(8.80E-12) | 8.13<br>17400 | 5.37E-14<br>(6.83E-13) | 7.31<br>15800 |
| rate | $k_0 + J = 8$ | | $k_0 + kJ = 8$ | | $k_0 + kJ = 8$ | |

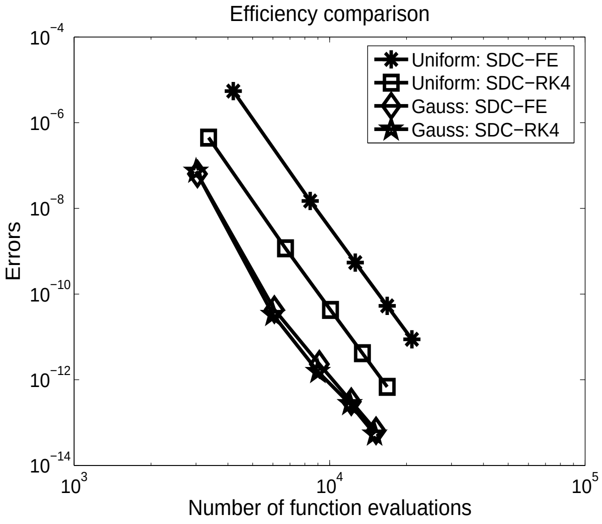

Figure 1: Efficiency comparison for (3.4)-(3.5) using the modified SDC method with 5 Legendre-Gauss quadrature nodes and the SDC method in [2] with 8 uniform nodes. Here function evaluations for the modified SDC method (Algorithm 2.2) include those for the original SDC and the additional ones for evaluating (2.28).

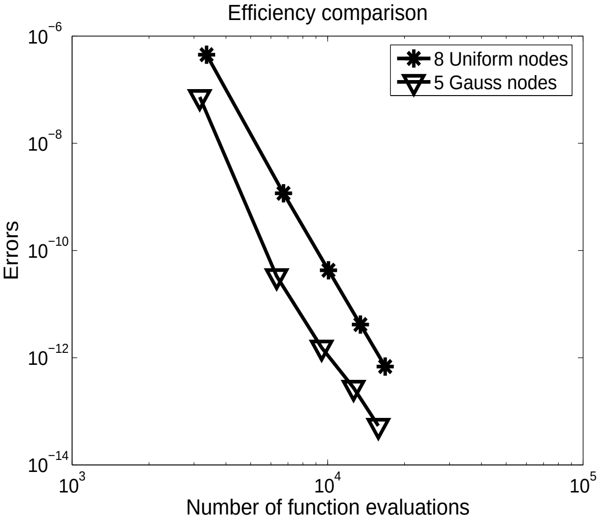

Figure 2: `Efficiency comparison between the modified SDC method on 5 Legendre-Gauss quadrature nodes and the original SDC method in [2] on 8 uniform nodes. In both cases,` $k_0 = k = 4$ `and` $J = 1$. `Here function evaluations for the modified SDC method (Algorithm 2.2) include those for the original SDC and the additional ones for evaluating (2.28).`

the amplification factor for a numerical method, $Am(\lambda)$, can be interpreted as the numerical solution after one time step of size 1. The stability region $S$ for a numerical method is the subset of the complex plane $\mathbb{C}$, consisting of all $\lambda$ such that $Am(\lambda) \le 1$, i.e.,

$$S = \{\lambda : Am(\lambda) \le 1\}. \tag{4.2}$$

In Figure 3(a), we plot the stability regions for the modified SDC methods using 8 Chebyshev-Lobatto quadrature nodes with FE, RK2 and RK4 in the prediction and correction steps. It is observed that the stability regions increase in size as higher-order numerical integrators are employed. A similar observation is made in Figure 3(b) when 12 Chebyshev-Lobatto quadrature nodes are used. In this case, we find an interesting phenomena that the stability region for SDC-RK3 is not contained in that of SDC-RK4.

It is pointed out that the main contribution of this article is to provide a convergence analysis for the SDC methods based on the integral operators. The convergence enhancement for general quadrature nodes is also based on the convergence analysis, in particular by obtaining the estimate (2.26). Our method is different from the one in [2] which uses the concept of *smoothness of the rescaled error vector* and the one in [5] which is based on the classical Stetter-Lindberg-Skeel framework and Spijker-type norms.

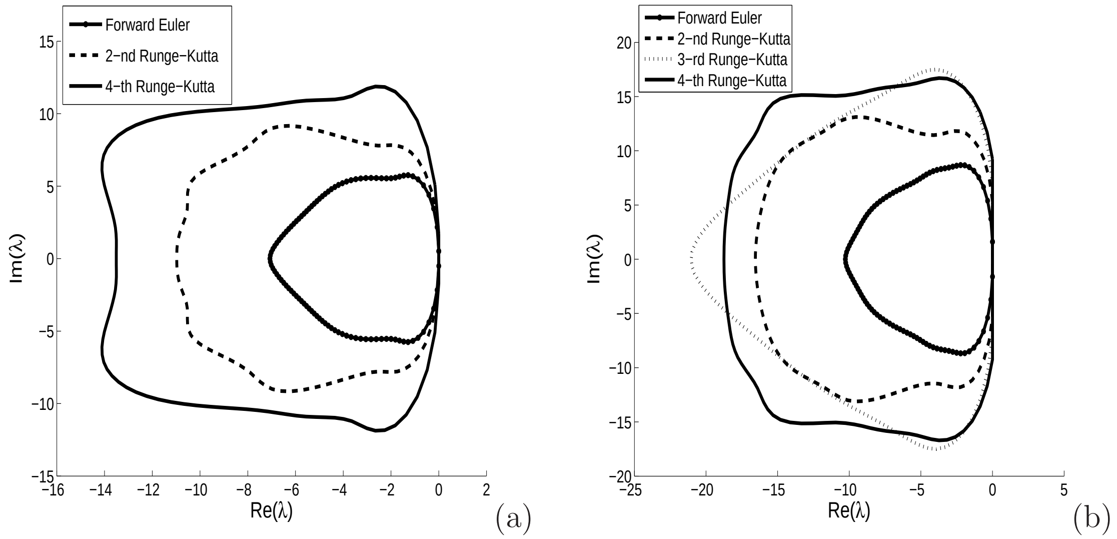

Figure 3: Stability regions for SDC methods with (a): 8 and (b): 12 Chebyshev-Lobatto quadrature nodes. In (a), more loops are used for lower order integrators so that all methods have the same convergence order 8. In (b), more loops are used for lower order integrators so that all methods have the same convergence order 12.

**Acknowledgments.** The first author thanks the useful discussions with Professors Chi-Wang Shu, John Strain and Jing-Mei Qiu. We are also grateful for Prof. Qiu for kindly providing us some relevant codes. This research was supported by Hong Kong Research Grants Council and Hong Kong Baptist University. The second author is supported by the Croucher Foundation of Hong Kong, the National Nature Science Foundation of China (11001259), the National Center for Mathematics and Interdisciplinary Science, CAS and the President Foundation of AMSS-CAS. The third author is supported by the National Science Foundation of China (11126215).

# References

[1] H. Brunner, Collocation Methods for Volterra Integral and Related Functional Equations, Cambridge University Press, 2004.

[2] A. Christlieb, B. Ong and J. Qiu, Spectral deferred correction methods with high order Runge-Kutta schemes in prediction and correction steps, Communications of Applied and Computational Mathematics, 4 (2009), 27-56.

[3] A. Christlieb, B. Ong and J. Qiu, A comment on high order corrections within spectral deferred correction methods, Math. of Comp., 79 (2010), 761-783.

[4] A. Dutt, L. Greengard and V. Rokhlin, Spectral deferred correction methods for ordinary differential equations, BIT Numerical Mathematics, 40(2) (2000), 241-266.

[5] A. Hansen and J. Strain, On the order of deferred correction, Appl. Numer. Math., 61(8) (2011), 961-973.

[6] A. T. Layton, On the choice of correctors for semi-implicit Picard deferred correction methods, Appl. Numer. Math. 58(6) (2008), 845-858.

[7] A. T. Layton and M. L. Minion, Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations, BIT Numerical Mathematics, 45 (2005), 341-373.

[8] A. T. Layton and M. L. Minion, Implications of the choice of predictors for semi-implicit Picard integral defferred correction methods, Comm. Appl. Math. Comput. Sci., 2(1) (2006), 1-34.

[9] R. D. Skell, A theoretical framework for proving accuracy results for deferred corrections, SIAM J. Numer. Anal., 19 (1982), 171-196.

[10] H. J. Stetter, Analysis of Discretization Methods for ODEs, Springer Verlag, Berlin, 1973.

[11] T. Tang and X. Xu, Accuracy enhancement using spectral postprocessing for differential equations and integral equations, Communication in Computational Physics, 5(2-4) (2009), 779-792.